SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD A104418

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A104418	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) DISTRIBUTED COMPUTATION OF FIXED POINTS		5. TYPE OF REPORT & PERIOD COVERED Technical Paper
		6. PERFORMING ORG. REPORT NUMBER LIDS-P-1135
7. AUTHOR(s) Dimitri P. Bertsekas		8. CONTRACT OR GRANT NUMBER(s) ARPA Grant ~~ONR~~-N00014-75-C-1183
9. PERFORMING ORGANIZATION NAME AND ADDRESS M.I.T. Laboratory for Information and Decision Systems Cambridge, MA 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Code No. 5T10 ONR Identifying No. 049-383
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209		12. REPORT DATE August 1981
		13. NUMBER OF PAGES 15
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Office of Naval Research Information Systems Program Code 437 Arlington, Virginia 22217		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

12 6 9 14 15 11 12 18

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

DTIC

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

We present an algorithmic model for distributed computation of fixed points whereby several processors participate simultaneously in the calculations while exchanging information via communication links. We place essentially no assumptions on the orderingof computation and communication between processors thereby allowing for completely uncoordinated execution. We provide a general convergence theorem for algorithms of this type, and demonstrate its applicability to several classes of problems including the calculation of fixed points of contraction and monotone mappings arising in linear and nonlinear systems of

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

410150

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DTIC FILE COPY

81 9 21 065

equations, shortest path problems, and dynamic programming.

Accession For
NTIS GRA&I ☒
DTIC TAB
Unannounced
Justification

By
Distribution/
Availability Codes

Dist	Avail and/or Special
A	

August 1981 LIDS-P-1135

DISTRIBUTED COMPUTATION OF FIXED POINTS*

by

Dimitri P. Bertsekas**

Abstract

We present an algorithmic model for distributed computation of fixed points whereby several processors participate simultaneously in the calculations while exchanging information via communication links. We place essentially no assumptions on the ordering of computation and communication between processors thereby allowing for completely uncoordinated execution. We provide a general convergence theorem for algorithms of this type, and demonstrate its applicability to several classes of problems including the calculation of fixed points of contraction and monotone mappings arising in linear and nonlinear systems of equations, shortest path problems, and dynamic programming.

*This research was conducted at the M.I.T. Laboratory for Information and Decision Systems with partial support provided by the Defense Advanced Research Projects Agency under Grant No. ONR-N00014-75-C-1183.

**Room No. 35-210, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139.

1. Introduction

There is presently a great deal of interest in distributed implementations of various iterative algorithms whereby the computational load is shared by several processors while coordination is maintained by information exchange via communication links. In most of the work done in this area the starting point is some iterative algorithm which is guaranteed to converge to the correct solution under the usual circumstances of centralized computation in a single processor. The computational load of the typical iteration is then divided in some way between the available processors, and it is assumed that the processors exchange all necessary information regarding the outcomes of the current iteration before a new iteration can begin.

The mode of operation described above may be termed synchronous in the sense that each processor must complete its assigned portion of an iteration and communicate the results to every other processor before a new iteration can begin. This assumption certainly enchances the orderly operation of the algorithm and greatly simplifies the convergence analysis. On the other hand synchronous distributed algorithms also have some obvious implementation disadvantages such as the need for an algorithm initiation and iteration synchronization protocol. Furthermore the speed of computation is limited to that of the slowest processor. It is thus interesting to consider algorithms that can tolerate a more flexible ordering of computation and communication between processors. Such algorithms have so far found applications in computer communication networks such as the ARPANET [1] where processor failures are common and it is quite complicated to maintain synchronization between the nodes of the entire network as they execute real-time network functions such as the routing algorithm.

In this paper we consider an extreme model of uncoordinated distributed algorithms whereby computation and communication is performed at the various processors completely independently of the progress in other processors. Perhaps somewhat surprisingly we find that even under these potentially chaotic circumstances of uncoordinated computation it is possible to solve correctly broad and significant classes of fixed point problems. A general convergence theorem is developed for this purpose which delineates circumstances under which convergence is guaranteed. The theorem is then applied to broad classes of fixed point problems involving contraction and monotone mappings.

2. A Model for Distributed Uncoordinated Fixed Point Algorithms

The fixed point problem considered in this paper is defined in terms of a set X, a class F of functions mapping X into the extended real line $[-\infty,+\infty]$, and a mapping T which maps F into itself. We wish to find an element J^* of F such that

$$J^* = T(J^*). \tag{1}$$

or equivalently

$$J^*(x) = T(J^*)(x), \qquad \forall\, x\varepsilon X, \tag{2}$$

where $J^*(x)$ and $T(J^*)(x)$ denote the values of the functions J^* and $T(J^*)$ respectively at the typical element $x\varepsilon X$. We will assume throughout that T has a unique fixed point J^* within the set F.

We provide some examples:

Example 1: (Fixed points of mappings on R^n). Let X be the finite set

$$X = \{1,2,\ldots,n\},$$

and F be the set of all real-valued functions on X. Then F can be identified with the n-dimensional space R^n in the sense that with each $J\varepsilon F$ we can associate the n-dimensional vector $\{J(1),J(2),\ldots,J(n)\}$. Similarly $T(J)$ can be identified with the n-dimensional vector $T(J)(1),\ldots,T(J)(n)$, so the fixed point problem (1) amounts to solving the system of n equations

$$J^* = T(J^*) \quad \text{or} \quad J^*(i) = T(J^*)(i), \quad \forall i = 1,\ldots,n \tag{3}$$

with the n unknowns $J^*(1),\ldots,J^*(n)$. It is also evident that any system of n (possibly nonlinear) equations with n unknowns can be formulated into a fixed point problem such as (3).

Example 2: (Shortest path problems). Let (N,L) be a directed graph where $N = \{1,2,\ldots,n\}$ denotes the set of nodes and L denotes the set of links. Let $N(i)$ denote the downstream neighbors of node i, i.e., the set of nodes j for which (i,j) is a link. Assume that each link (i,j) is assigned a positive scalar a_{ij} referred to as its length. Assume also that there is a directed path to node 1 from every other node. Then it is known ([2], p.67) that the shortest path distances $J^*(i)$ to node 1 from all other nodes i solve uniquely the equations

$$J^*(i) = \min_{j\varepsilon N(i)} \{a_{ij} + J^*(j)\} , \quad \forall i \neq 1 \tag{4a}$$

$$J^*(1) = 0 \tag{4b}$$

If we make the identifications $X = \{1,2,\ldots,n\}$, F: Set of all functions mapping X into $[0,+\infty]$, and define $T(J)$ for all $J\varepsilon F$ by means of

$$T(J)(i) = \begin{cases} \min_{j \in N(i)} \{a_{ij} + J(j)\} & \text{if } i \neq 1 \\ 0 & \text{if } i = 1 \end{cases} \quad (5)$$

then we find that the fixed point problem (2) reduces to the shortest path problem.

The shortest path problem above is representative of a broad class of dynamic programming problems which can be viewed as special cases of the fixed point problem (2) and can be correctly solved by using the distributed algorithms of this paper (see [3]).

Our algorithmic model can be described in terms of a collection of n computation centers (or processors) referred to as nodes and denoted 1,2,...,n. The set X is partitioned into n disjoint sets denoted $X_1,\ldots,X_n$, i.e.

$$X = \bigcup_{i=1}^{n} X_i \ , \quad X_i \cap X_j = \emptyset, \quad \text{if } i \neq j.$$

Each node i is assigned the responsibility of computing the values of the solution function J* [c.f. (1),(2)] at all $x \varepsilon X_i$.

At each time instant, node i can be in one of three possible states compute, transmit, or idle. In the compute state node i computes a new estimate of the values of the solution function J* for all $x \varepsilon X_i$. In the transmit state node i communicates the estimate obtained from the latest computation to one or more nodes j ($j \neq i$). In the idle state node i does nothing related to the solution of the problem. It is assumed that a node can receive a transmission from other nodes simultaneously with computing or transmitting, but this is not a real restriction since, if needed, a time period in a separate receive state can be lumped into a time period in the idle state.

We assume that computation and transmission for each node takes place in uninterupted time intervals $[t_1,t_2]$ with $t_1 < t_2$, but do not exclude the possibility that a node may be simultaneously transmitting to more than one nodes nor do we assume that the transmission intervals to these nodes have the same origin and/or termination. We also make no assumptions on the length, timing and sequencing of computation and transmission intervals other than the following:

Assumption (A): There exists a positive scalar P such that, for every node i, every time interval of length P contains at least one computation interval for i and at least one transmission interval from i to each node $j \neq i$.

Each node i also has a buffer B_{ij} for each $j \neq i$ where it stores the latest transmission from j, as well as a buffer B_{ii} where it stores its own estimate of values of the solution function for all $x \in X_i$. The contents of each buffer B_{ij} at time t are denoted J_{ij}^t. Thus J_{ij}^t is, for every t, a function from X_j into $[-\infty,\infty]$ and may be viewed as the estimate by node i of the restriction of the solution function J^* on X_j available at time t. The rules according to which the functions J_{ij}^t are updated are as follows:

1) If $[t_1,t_2]$ is a transmission interval from node j to node i the contents $J_{jj}^{t_1}$ of the buffer B_{jj} at time t_1 are transmitted and entered in the buffer B_{ij} at time t_2, i.e.

$$J_{ij}^{t_2} = J_{jj}^{t_1}. \tag{6}$$

2) If $[t_1,t_2]$ is a computation interval for node i the contents of buffer B_{ii} at time t_2 are replaced by the restriction of the function $T(J_i^{t_1})$ on X_i where, for all t, J_i^t is defined by

$$J_i^t(x) = \begin{cases} J_{ii}^t(x) & \text{if } x \varepsilon X_i \\ J_{ij}^t(x) & \text{if } x \varepsilon X_j, \quad j \neq i \end{cases} \tag{7}$$

In other words we have

$$J_{ii}^{t_2}(x) = T(J_i^{t_1})(x), \quad \forall x \varepsilon X_i \tag{8}$$

3) The contents of a buffer B_{ii} can change only at the end of a computation interval for node i. The contents of a buffer B_{ij}, $j \neq i$ can change only at the end of a transmission interval from j to i.

Our objective is to derive conditions under which

$$\lim_{t\to\infty} J_i^t(x) = J^*(x), \quad \forall x \varepsilon X_i,\ i = 1,2,\ldots,n, \tag{9}$$

and J* is the unique fixed point of T within F [cf. (1),(2)]. This is the subject of the next section.

3. A General Convergence Theorem

We formulate the following assumption under which we will subsequently prove convergence of the type indicated in (9). In what follows $\overline{F}$ denotes the set of all functions from F into $[-\infty,+\infty]$, F x F denotes the Cartesian product of the set of functions F with itself, and F^n denotes the Cartesian product of F with itself n times.

Assumption B: There exists a sequence $\{F_k\}$ of subsets of F with the following properties:

a) If $\{J_k\}$ is a sequence in F such that $J_k \varepsilon F_k$ for all k then

$$\lim_{k\to\infty} J_k(x) = J^*(x) \quad \forall x \varepsilon X. \tag{10}$$

b) For all $k = 0,1,\ldots$ and $i = 1,\ldots,n$

$$J \varepsilon F_k \Longrightarrow T_i(J) \varepsilon F_k \tag{11}$$

where $T_i : F \to \overline{F}$ is the mapping defined by

$$T_i(J)(x) = \begin{cases} J(x) & \text{if } x \not\varepsilon X_i \\ T(J)(x) & \text{if } x \varepsilon X_i. \end{cases} \tag{12}$$

c) For all $k = 0,1,\ldots$ and $j \varepsilon 1,\ldots,n$

$$J \varepsilon F_k,\ J' \varepsilon F_k \Longrightarrow C_j(J,J') \varepsilon F_k \tag{13}$$

where $C_j : F \times F \to \overline{F}$ is the mapping defined by

$$C_j(J,J')(x) = \begin{cases} J(x) & \text{if } x \not\varepsilon X_j \\ J'(x) & \text{if } x \varepsilon X_j \end{cases} \tag{14}$$

d) For all $k = 0,1,\ldots$

$$J_1 \varepsilon F_k, \ldots, J_n \varepsilon F_k \Longrightarrow \overline{T}(J_1,J_2,\ldots,J_n) \varepsilon F_{k+1} \tag{15}$$

where $\overline{T} : F^n \to \overline{F}$ is the mapping defined by

$$\overline{T}(J_1,J_2,\ldots,J_n)(x) = T(J_i)(x), \quad \forall x \varepsilon X_i,\ i = 1,\ldots,n \tag{16}$$

Assumption B seems rather complicated so it may be worth providing some motivation for introducing it. Property a) specifies how the sets F_k should relate to the solution J^*. Property d) guarantees that if the functions in the buffers of all nodes $i = 1,\ldots,n$ belong to F_k and a computation phase is carried out simultaneously at all nodes followed by a communication phase from every node to every other node then the resulting function in the buffer of each node (which will be the same for all nodes), will belong to F_{k+1}. Assumptions a) and d) alone guarantee that the algorithm will converge to the correct solution if executed in a synchronous manner, i.e., a simultaneous computation phase at all nodes is followed by a simultaneous communication phase from each node to all other nodes and the process is repeated. Property b) involves the mapping T_i which is related to a computation phase at node i [compare (7), (8) with (12)], while property c) involves the mapping C_j which is related to a communication phase from node j to some other node [compare (6) with (13), (14)]. Basically properties b) and c) guarantee that the sets F_k are closed with respect to individual node computation and communication. By this we mean that if all buffer contents are within F_k then after a single node computation or communication all buffer contents will remain in F_k. The following proposition asserts that when properties b) and c) hold in addition to a) and d), then the algorithm converges to the correct solution when operated in a totally uncoordinated manner.

Proposition: Let Assumptions A and B hold, and assume that the initial buffer contents J_i^o at each node i = 1,...,n belong to F_o. Then

$$\lim_{t\to\infty} J_i^t(x) = J^*(x), \quad \forall x\varepsilon X,\ i = 1,\dots,n \tag{17}$$

where J_i^t is defined by (7) for all $t \geq 0$ and i = 1,...,n.

Proof: We will show that for every k = 0,1,... and $t \geq 0$ the condition

$$J_i^t \varepsilon F_k, \quad \forall i = 1,\dots,n \tag{18}$$

implies

$$J_i^{t'} \in F_k, \qquad \forall t' \geq t, \ i = 1,\ldots,n \tag{19}$$

$$J_i^{t'} \in F_{k+1}, \qquad \forall t' \geq t+2P, \ i = 1,\ldots,n \tag{20}$$

where P is the scalar of Assumption A. In view of condition a) of Assumption B, this will suffice to prove the proposition.

Assume that (18) holds for some $k = 0,1,\ldots$ and $t \geq 0$. Then (19) clearly holds since, for $t' \geq t$, the buffer content $J_i^{t'}$ of node i at t' is obtained from the buffer contents J_j^t of all nodes $j = 1,\ldots,n$ at t via operations that (according to conditions b) and c) of Assumption B) preserve membership in F_k.

Consider the contents $J_{ii}^{t'}$ of the buffers B_{ii}, $i = 1,\ldots,n$ at a time $t' \geq t+P$. Since (by Assumption A) at least one computation was performed at each node i in the interval [t,t+P], we have that [cf. (7)]

$$J_{ii}^{t'}(x) = T(J_i^{\bar{t}})(x), \qquad \forall x \in X_i, \ i = 1,\ldots,n \tag{21}$$

where $J_i^{\bar{t}}$ is the buffer content of node i at some time $\bar{t} \in [t,t']$ ($\bar{t}$ depends on i), and by (19)

$$J_i^{\bar{t}} \in F_k, \qquad \forall i = 1,\ldots,n.$$

Since the interval [t+P, t+2P] contains at least one communication interval from every node to every other node it follows that for any $t' \geq t+2P$ each buffer B_{ij} contains a function $J_{ij}^{t'}$ such that [cf. (6), (21)]

$$J_{ij}^{t'}(x) = J_{jj}^{\tilde{t}}(x) = T(J_j^{\bar{t}})(x), \qquad \forall x \in X_j, \ i,j = 1,\ldots,n \tag{22}$$

where $J_{jj}^{\tilde{t}}$ is the content of buffer B_{jj} at node j at some time $\tilde{t} \in [t+P,t+2P]$

and $J_j^{\bar{t}}$ is the buffer content of node j at some time $\bar{t}\varepsilon[t,\tilde{t}]$. (Again here the times $\tilde{t}$ and $\bar{t}$ depend on i and j).

By using (22) and (19) we can assert that for each $t' \geq t+2P$ and $i = 1,\ldots,n$ there exist functions $\bar{J}_j \varepsilon F_k$, $j = 1,\ldots,n$ such that

$$J_i^{t'}(x) = T(\bar{J}_j)(x), \qquad x\varepsilon X_j, \ i = 1,\ldots,n.$$

It follows from condition d) of Assumption B [cf. (15),(16)] that

$$J_i^{t'} \varepsilon F_{k+1}, \qquad \forall t' \geq t+2P, \ i = 1,\ldots,n$$

which is (20). This completes the proof of the proposition. Q.E.D.

Note that (18) and (20) can form the basis for an estimate of the rate of convergence of the algorithm. For example if there exists an index $\bar{k}$ such that $F_k = F_{\bar{k}} = \{J^*\}$ for all $k \geq \bar{k}$ (i.e. after some index the sets F_k contains only one element--the solution function J^*), then it follows from (18)-(20) that the distributed algorithm converges to the correct solution in a finite amount of time. This argument can, for example, be used to establish finite time convergence for the distributed algorithm as applied to the shortest path problem of Section 2.

4. Special Cases

In this section we verify that Assumption B of the previous section is satisfied for some important classes of problems.

Contraction Mappings with Respect to the Sup-Norm

Let $\bar{F}$ be the space of all bounded real-valued functions on X and assume that $T(J)\varepsilon\bar{F}$ for all $J\varepsilon F$. Let $||\cdot||$ denote the sup-norm on this space, i.e.

$$||J|| = \sup_{x\varepsilon X} |J(x)|, \qquad \forall x\varepsilon X. \tag{23}$$

Assume that F is a closed sphere centered at J*. Then F is complete ([4]), so if there exists a scalar α with $0 < \alpha < 1$ such that the mapping T satisfies

$$||T(J) - T(J')|| \leq \alpha||J-J'||, \qquad \forall J,J'\varepsilon F$$

there T has a unique fixed point in F denoted J* [4].

For q > 0 define

$$F_k = \{J\varepsilon\bar{F} \mid ||J-J^*|| \leq \alpha^k q\}, \quad k = 0,1,\ldots$$

It is evident that if $F_c \subset F$ then the sequence $\{F_k\}$ satisfies conditions a)-d) of Assumption B.

We note that the use of the sup-norm (23) is essential in order for Assumption B to hold and that if T is a contraction mapping with respect to some other norm then Assumption B need not be satisfied.

An important example where T is a contraction mapping with respect to the sup-norm arises in Newton's method for solving nonlinear systems of equations. Let $G: R^n \to R^n$ be a continuously differentiable mapping and assume that $J^*\varepsilon R^n$ satisfies $G(J^*) = 0$ and that the nxn Jacobian $\frac{\partial G(J^*)}{\partial J}$ is nonsingular at J*.

Then it can be shown that the Newton mapping

$$T(J) = J - \left[\frac{\partial G(J)}{\partial J}\right]^{-1} G(J)$$

is a contraction mapping with respect to the sup-norm within a neighborhood of J* and therefore satisfies Assumption B within that neighborhood.

P-Contraction Mappings

Let $X = \{1,2,\ldots,n\}$ and $X_i = \{i\}$. Then functions J can be identified with the corresponding n-dimensional vectors $\{J(1),\ldots,J(n)\}$ and F can be identified with a subset of R^n. For each $J = \{J(1),\ldots J(n)\}$ we denote by $|J|$ the vector $\{|J(1)|,\ldots,|J(n)|\}$. Suppose that T is such that

$$|T(J) - T(J')| \leq P\,|J-J'| \quad , \forall J,J' \varepsilon R^n \tag{24}$$

where P is a substochastic matrix, i.e. all elements of P are nonnegative and the sum of the elements of each row of P is less than or equal to unity, and the inequality (24) is meant to hold componentwise. Assume further that $\lim_{k\to\infty} P^k = 0$. Then it can be shown that T has a unique fixed point J* in R^n ([5], p.433).

For any $q > 0$ define

$$F_k = \{J \varepsilon R^n | \; |J-J^*| \leq q\, P^k e\}$$

where $e = \{1,1,\ldots,1\}$ is the unit vector in R^n. Then it can easily be seen again that the sequence $\{F_k\}$ satisfies conditions a)-d) of Assumption B.

Fixed point problems involving P-contraction mappings arise in dynamic programming ([6], p.374) and solution of systems of nonlinear equations ([5], Section 13.1).

Monotone Mappings

Assume that T has the monotonicity property

$$J \epsilon F,\ J' \epsilon F,\ J(x) \leq J'(x),\ \forall x \epsilon X \Rightarrow T(J)(x) \leq T(J')(x),\ \forall x \epsilon X \tag{25}$$

Assume further that there exist two functions $\underline{J}$ and $\overline{J}$ in F such that

$$\{J \mid \underline{J}(x) \leq J(x) \leq \overline{J}(x),\ \forall x \epsilon X\} \subset F \tag{26}$$

and for all k = 0,1,...

$$T^k(\underline{J})(x) \leq T^{k+1}(\underline{J})(x) \leq T^{k+1}(\overline{J})(x) \leq T^k(\overline{J})(x),\quad \forall x \epsilon X \tag{27}$$

and

$$\lim_{k\to\infty} T^k(\underline{J})(x) = \lim_{k\to\infty} T^k(\overline{J})(x) = J^*(x),\ \forall x \epsilon X. \tag{28}$$

As an example consider the shortest path problem of Example 2 in Section 2, and the functions

$$\underline{J}(i) = 0, \quad \forall i = 1,\ldots,n$$

$$\overline{J}(i) = \begin{cases} \infty & \text{if } i \neq 1 \\ 0 & \text{if } i = 1. \end{cases}$$

It is easily verified that the corresponding mapping T satisfies (25) and that $\underline{J}$, $\overline{J}$ as defined above satisfy (26), (27), (28).

Define now for k = 0,1,...

$$F_k = \{J \mid T^k(\underline{J})(x) \leq J(x) \leq T^k(\overline{J})(x),\ x \epsilon X\}.$$

Then it is easily seen that the sequence $\{F_k\}$ satisfies conditions a)-d) of Assumption B.

Fixed point problems involving monotone mappings satisfying (25) arise in dynamic programming [3], [6], [7] and solution of systems of nonlinear equations ([5], Section 13.2).

5. Conclusions

The analysis of this paper shows that broad classes of fixed point problems can be solved by distributed algorithms that operate under very weak restrictions on the timing and ordering of processor computation and communication phases. It is also interesting that the initial processor buffer contents need not be identical and can vary within a broad range. This means that for problems that are being solved continuously in real time it is not necessary to reset the initial conditions and resynchronize the algorithm each time the problem data changes. As a result the potential for tracking slow changes in the solution function is improved and algorithmic implementation is greatly simplified.

References

[1] J. McQuillan, G. Falk, and I. Richer, "A Review of the Development and Performance of the ARPANET Routing Algorithm", IEEE Trans. on Communications, Vol. COM-26, 1978, pp. 1802-1811.

[2] E. L. Lawler, Combinatorial Optimization: Networks and Matroids, Holt, Rinehart, and Winston, N.Y., 1976.

[3] D. P. Bertsekas, "Distributed Dynamic Programming", Report LIDS-P-1060, Laboratory for Information and Decision Systems, Mass. Institute of Technology, Cambridge, Mass., Dec. 1980 (to appear in IEEE Trans. on Automatic Control).

[4] H. L. Royden, Real Analysis, McMillan, N.Y., 1963.

[5] J. M. Ortega and W. C. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, N.Y., 1970.

[6] D. P. Bertsekas, Dynamic Programming and Stochastic Control, Academic Press, N.Y., 1976.

[7] D. P. Bertsekas and S. E. Shreve, Stochastic Optimal Control: The Discrete Time Case, Academic Press, N.Y., 1978.